Dynamic constraint generation in *HASTUS-CrewOpt,* a column generation approach for transit crew scheduling

By Alain Dallaire, Charles Fleurent, and Jean-Marc Rousseau

Introduction

Transit crew scheduling is a challenging practical optimization problem that has stimulated a lot of research activity over the past three decades. Although several algorithmic approaches to this problem have been tried over the years (see [7]), column generation is generally considered the most powerful method currently available (see [1, 3]). First used commercially in Public Transport by GIRO (see [2, 3, 6]), column generation was initially restricted to problems of small to moderate size. Since then, significant developments have enabled it to solve problems of all sizes efficiently, leading GIRO to select *CrewOpt* as the standard crew scheduling algorithm for all our clients (see [3]).

Now that problem size is not as critical as it once was, GIRO has devoted a great deal of research activity to adding new capabilities to the algorithm. For instance, although *CrewOpt* has been able to obtain less costly crew solutions by changing trip links from source vehicle schedules for many years, it can now do so more efficiently. This approach is in keeping with a recent research trend aimed at designing optimization methods that integrate vehicle and crew scheduling (see [4, 5]). Another recent feature of *CrewOpt* is its ability to generate cost-effective solutions that respect the characteristics of an existing duty set as much as possible, so changes to the scheduling environment can be accommodated with minimal disruptions to previous crew schedules.

These new features are well received by schedulers because they make it possible to produce solutions that are both less costly and more reliable from an operational perspective. However, the new capabilities also induce larger optimization problems that make them more difficult to solve efficiently. For instance, when modifications to links between trips are permitted, a large number of linear constraints must be included in the mathematical model to limit increases to the number of vehicles that operate during peak hours and to ensure the new trip links are feasible for all duties. If many trips, depots, and vehicle groups are involved, the number of constraints can reach thousands. Likewise, when similar solutions are desired, linear constraints must be added to prevent creating duties that are similar to more than one duty from the original crew schedule.

In the next section, we explain how both goals can be integrated into a set-partitioning model with additional constraints that can be handled by column generation. Section 3 focuses on the practical implementation of this model when solving large-scale problems, including the dynamic generation of constraints and strategies used to accelerate the process. In Section 4, numerical results are provided for some real applications, while section 5 concludes this paper.

Some advanced features of *CrewOpt*

Basic model

The transit crew scheduling problem consists of identifying an optimal set of duties that completely covers a given vehicle schedule, while respecting several constraints. Although a lot of data is needed to completely describe a practical problem (relief opportunities, walking times for drivers, collective agreements, etc.), it can basically be formulated as a set-partitioning problem (see [2] for instance). Out of the set all possible duties, the lowest-cost subset of duties that covers each driving task exactly once (i.e. indivisible portion of driving between consecutive relief points) must be identified. In practice, several additional constraints must also be taken into account to control characteristics of individual duties and of the solution as a whole.

A general model can be summarized as follows.

$$
\begin{aligned}
\text{Min } & \sum_{i\in I} c_i x_i + \sum_{k\in L} f_k^- s_k^- + \sum_{k\in L} f_k^+ s_k^+ \\
\text{s.t. } & \sum_{i\in I} a_{ij} x_i = 1, \quad j \in J \\
& \sum_{i\in I} b_{ik} x_i + s_k^- + s_k^+ = d_k, \quad k \in L \qquad\qquad (P) \\
& x_i \in \{0,1\} \\
& s_k^- \geq 0, \quad k \in L \\
& s_k^+ \geq 0, \quad k \in L
\end{aligned}
$$

Where:

I is the set of all possible duties;
J is the set of all driving tasks;
L is the set of additional constraints;
c_i is the cost of duty $i \in I$;

$$a_{ij} = \begin{cases} 1 \text{ if duty } i \text{ covers driving task } j,\ i \in I,\ j \in J \\ 0 \text{ otherwise;} \end{cases}$$

b_{ik} is the coefficient of variable i in additional constraint $k \in L$;

f_k^- is the cost of the slack variable of additional constraint $k \in L$;

f_k^+ is the cost of the surplus variable of additional constraint $k \in L$;

s_k^- is the slack variable of additional constraint $k \in L$;

s_k^+ is the surplus variable of additional constraint $k \in L$;

d_k is the right-hand side value of additional constraint $k \in L$;

$$x_i = \begin{cases} 1 \text{ if duty } i \text{ is selected in the solution, } i \in I \\ 0 \text{ otherwise.} \end{cases}$$

In the objective function, the first term represents the sum of the costs of selected duties, while the second and third terms correspond to the sum of costs of slack and surplus variables from the additional constraints. The first set of constraints specifies that each driving task must be covered by exactly one duty, while the second set includes linear constraints that describe requirements for a specific application context.

Although the problem (P) correctly describes transit crew scheduling problems, it cannot be solved directly in practice, as the cardinality of set I can easily reach millions for problems of typical size. In our column generation approach, a master linear program similar to (P) is repeatedly solved for subsets $I' \subset I$ in which I' are much smaller sets. These subsets of duties are generated by heuristically solving constrained shortest path problems (auxiliary problems) in several networks, where costs on arcs come from duty costs and dual values obtained from the resolution of the previous master problem. Constraints that apply to individual duties are handled when the auxiliary problems are solved, so only valid duties are generated. Restrictions that apply to the global solution are formulated as side constraints in the master problem.

The column generation method is now well known and has been widely studied and used in academia. It is, however, relatively complex and in practical contexts the efficiency of its implementation is critical. *CrewOpt* has the distinct advantage of using the GENCOL package as the core of its the column generation code. GENCOL is the product of more than 20 years of ongoing research on these techniques at GERAD (Groupe d'Étude et de Recherche en Analyse des Décisions) and includes several mechanisms that contribute to its performance.

Integrated bus and duty scheduling

It is clear that simultaneously scheduling vehicles and duties provides flexibility that can lead to significant savings in overall solution costs. For this reason, GIRO's crew scheduling algorithms have always offered the ability to revise the source vehicle schedule while duties are optimized. Our former heuristic algorithm SuperMicro (see [7]), which was used by many transit companies during the eighties and early nineties, could change existing links in the source vehicle schedule when such revisions led to less costly solutions. Since there generally are fewer constraints on vehicles than on duties, the resulting vehicle schedules often remained acceptable, even though the only measures used to protect them were penalties on changes to trip links during the peak hours (when such modifications are more likely to increase the required number of vehicles).

When *CrewOpt* was first introduced, GIRO made the adaptations necessary to permit revisions to the source vehicle schedule. These adaptations essentially consisted of modifying the structure of the network used by the auxiliary problem so the duties generated can implicitly include changes to trip links, and adding linear constraints to the master problem so decisions affecting trip links and driver reliefs are compatible. However, we observed that for *CrewOpt*, penalizing modifications to trip links during peak hours was not always an effective way of limiting increases in vehicle usage. In some instances, because of its accuracy, the column generation method produced solutions that avoided using penalized relief points yet still increased the number of vehicles to reduce the overall cost of duties. We thus decided to include a set of additional linear constraints in our model to forbid increases in the number of vehicles during some periods of the day. Information required to generate these constraints in the master problems can be extracted from the duties that are considered at a particular iteration.

More recently, we introduced the option of adding even more constraints to prevent increases to the number of vehicles by depot and by vehicle group in multiple-depot scheduling contexts. These constraints can be directly added to the master problem as long as some restrictions are imposed on the structure of duties generated when solving the auxiliary problem. Since we usually start from a previously optimized source vehicle schedule, we simply forbid the introduction of duties if they include new trip links that would lead to a change of depot or vehicle group in the source vehicle solution.

In the scientific literature, there are two main approaches to model and solve integrated vehicle and crew scheduling problems. The first approach (see [4]) uses a mixed model that includes network flow variables normally associated with single-depot vehicle scheduling problems, binary variables associated to a set partitioning component for the selection of duties, and additional constraints that bind both sets of variables into a compatible integrated solution. Although this model has some theoretical interest, it has not yet led to solution methods that can handle large-scale practical problems. The second approach (see [5]) relies on a set-partitioning model that is essentially similar to what *CrewOpt* uses.

Although *CrewOpt* can simultaneously generate an integrated vehicle/crew schedule from scratch, in practice we find it is much better to first generate an optimized vehicle schedule to use as a starting point for a subsequent phase. The advantages of such an approach are significant when specialized algorithms with more features are used to generate the initial vehicle solutions. For example, GIRO's *Minbus* algorithm includes many advanced options that permit multiple depot/vehicle-group scheduling, handling of complex constraints, use of midday parking, and controlled deviations to minimum layover and deadhead durations. It can also permit trip shifting (minor shifts to trip times), which uses an integrated timetabling/vehicle scheduling approach that can lead to substantial savings in the final solution.

Thus as long as an efficient, state-of-the-art vehicle scheduling algorithm is available, there is nothing to lose, but rather much to gain, by generating the best possible vehicle schedule as a starting point for crew scheduling.

Generating similar crew scheduling solutions

An algorithm that can generate a cost-effective crew solution that is similar to an initial crew schedule can also have important practical benefits. Among other things, it can lead to improved operational reliability and reduced training needs if similar duties are consistently proposed to drivers. One way of achieving this goal is by defining a distance function to evaluate quantitatively how similar two duties are, and reducing the cost of some generated duties when they are considered similar to one in the starting solution. Several factors can be considered in the evaluation of the distance of two duties, including their crew bases and duty types, number of breaks, percentage of common driving time, etc. When the algorithm generates a duty that is considered similar, it is also important to associate it to a counterpart in the starting solution. This makes it possible to keep the same identifiers so these duties can be recognized and assigned to the same drivers when possible.

Additional linear constraints must also be added to the master problem (P), to ensure that no more than one duty is associated to each duty in the starting solution.

Handling large numbers of constraints

In many of the older heuristic approaches to crew scheduling (see [7]), constraint violations had to be penalized in the objective function, and penalty values had to be adjusted with Lagrangean methods that were not always robust. An important advantage of the column generation method resides in its ability to directly handle linear constraints that can be added to a master problem, which is solved by linear programming. However, when revisions to the vehicle schedule are permitted or a similar solution is sought, the number of constraints can grow and become a critical issue in the resolution process.

We find that directly adding a large number of linear constraints to the master problem is generally not worthwhile because it makes the optimization process less efficient. First, the linear programs are larger and take more time to solve. Second, we observe that better solutions are often found when constraints are added progressively. Although we cannot yet provide a definitive explanation for this behavior, our tests indicate that this strategy should be preferred in most cases. In our implementation, some additional constraints are initially ignored and progressively added as violations are detected. For instance, vehicle count constraints can be disregarded in the early stages, so the algorithm can first concentrate on generating efficient duties that cover all driving tasks at low cost. Once a good solution to the continuous relaxation of the master problem is found, vehicles can be counted, and the appropriate constraints are included in the subsequent master problems to be solved. To avoid fixing fractional variables to integer values when a constraint violation occurs, adding constraints has priority over any other branching decision.

In addition, several strategies are used to accelerate the resolution process. These include a variety of branching schemes to control the exploration of the branch and bound tree. It is possible, for example, to branch on the decision to link individual driving tasks into a solution or the decision to include a full duty. Many parameters also control threshold values for which some variables will be fixed, as well as the number and characteristics of variables that will be fixed at each node of the branch and bound tree. Most of these strategies are part of the GENCOL package and have been fine-tuned by GIRO over the years to yield the best results for transit scheduling problems.

Results

CrewOpt is now GIRO's standard crew scheduling optimization algorithm and is repeatedly used by the majority of our clients (Montréal, New York, Chicago, Barcelona, Lyon, Turin, Rotterdam, Geneva, Vienna, Hamburg, Canberra, to name a few). This wide acceptance enables us to test our developments on a variety of environments and identify robust implementations. For instance, we found that the dynamic constraint generation mechanism described in the previous section consistently improves results, either by accelerating the resolution process or by generating solutions of better quality. We believe that the inferior results obtained when all constraints are present from the start can be explained by the fact that dual information from a larger number of constraints has to be passed from the master to the auxiliary problems, which can lead the heuristic mechanisms to overlook critical duties. Although some improvements and parameter setting could alleviate this difficulty, dynamic constraint generation is a very effective and reliable technique in practice. To illustrate these findings, we provide typical results observed for real examples from two major European transit companies (tests were conducted on a 3 GHz Xeon processor with 1 GB of memory).

In the first example, we consider a problem involving 900 driving tasks. With this problem, more than 2000 additional constraints theoretically have to be added to control vehicle count increases. However, only about 50 of these are required when they are dynamically generated as violations are detected. In this case, a better solution is found and the process is significantly faster with dynamic constraint generation (see Table 1).

	All constraints always present	Dynamic constraint generation
Solution cost	53,461	52749 (1.3 % improvement)
# of additional constraints	2,037	52 (2.5 % of possible constraints)
Execution time (seconds)	11,803	3,815

Table 1 – Results for the first test problem.

In our second example (see Table 2), the problem instance involves 706 driving tasks and 795 potential additional constraints. In this case, dynamically generating constraints takes more time but produces a much better solution. The increase in execution time can be explained by the fact that a larger percentage of additional constraints had to be generated and that the algorithm spent more time generating new columns and optimizing. However, this turned out to be beneficial by providing more accurate dual information early in the process, and preventing heuristic mechanisms to overlook worthy duties.

	All constraints always present	Dynamic constraint Generation
Solution cost	48,412	47473 (1.9 % improvement)
# of additional constraints	795	239 (30 % of possible constraints)
Execution time (seconds)	6,683	13,063

Table 2 – Results for the second test problem.

Conclusion

Column generation is currently recognized as the best optimization approach for transit crew scheduling. As algorithmic advances have enabled us to tackle problems of large size, developments can now focus on adding new capabilities to the existing methods. Controlled revisions to vehicle schedules and the ability to generate crew solutions that are similar to a reference crew schedule are two examples of interesting features that are now available to schedulers. These options put some pressure on the resolution process as several additional constraints are added to the theoretical model. Fortunately, careful implementation can alleviate some of that pressure, and lead to better schedules that are currently put on the street every day.

References

[1] Borndörfer, R., Grötschel, M., Löbel, A. (2001). Scheduling Duties by Adaptive Column Generation. ZIB-Report 01-01, Konrad-Zuse-Zentrum für Informationstchnik, Berlin.[2] Desrochers, M., Soumis, F. (1989). A Column Generation Approach to the Urban Transit Crew Scheduling Problem. Transportation Science 23: 1-13.

[3] Fleurent, C., Séguin, L. (2002). *CrewOpt*: A Column Generation Method for Transit Crew Scheduling. Presented at the International Conference on Operations Research SOR 2002, University of Klagenfurt, Austria, September 2002.

[4] Freling, R., Wagelmans, APM., Paixao, JMP. (1999). An Overview of Models and Techniques for Integrating Vehicle and Crew Scheduling. In: Wilson NHM (ed), Computer-Aided Transit Scheduling, Lecture Notes in Economics and Mathematical Systems 471, Springer, Berlin, pp 441-460.

[5] Haase, K., Desaulniers, G., Desrosiers, J. (2001). Simultaneous Vehicle and Crew Scheduling in Urban Mass Transit Systems. Transportation Science 35: 286-303.

[6] Rousseau, J.-M., Desrosiers, J. (1995). Results Obtained with *CrewOpt*, a Column Generation Method for Transit Crew Scheduling. In: Daduna JR, Branco I, Paixao JMP (eds), Computer-Aided Scheduling of Public Transport, Lecture Notes in Economics and Mathematical Systems 430, Springer, Berlin, pp 349-358.

[7] Rousseau, J.-M., Wren, A. (1995). Bus Driver Scheduling – An Overview. In: Daduna JR, Branco I, Paixao JMP (eds), Computer-Aided Scheduling of Public Transport, Lecture Notes in Economics and Mathematical Systems 430, Springer, Berlin, pp 173-187.